CLAIMS

What is claimed is:

1. An executable code check system comprising:

an input component that receives an object file and a specification associated with the object file, the specification comprising information associated with a plug-in condition for a method; and,

a checker that employs the specification to facilitate static checking of the object file, the checker providing information if a fault condition is determined.

2. The system of claim 1, the plug-in condition comprising a precondition for the method.

3. The system of claim 2, the checker providing information associated with an object's state after a call to the method, the information being based, at least in part, upon the plug-in precondition.

4. The system of claim 1, the plug-in condition comprising a postcondition for the method.

5. The system of claim 4, the checker providing information associated with an object's state after a call to the method, the information being based, at least in part, upon the plug-in postcondition.

6. The system of claim 1, the object file being based, at least in part, upon a language that compiles to Common Language Runtime.

7. The system of claim 1, the object file being based, at least in part, upon at least one of C#, Visual Basic.net and Managed C++.

8. The system of claim 1, the specification comprising information associated with a state-machine protocol.

9. The system of claim 8, a state of an object modeled with a custom state.

10. The system of claim 9, the state of the object further being modeled with a custom state component.

11. The system of claim 10, the specification comprising at least one of a plug-in precondition and a plug-in postcondition method, which is a method of the custom state that is invoked by the checker to perform interface-specific state checks and state transitions.

12. The system of claim 1, wherein the specification is embedded with the object file.

13. The system of claim 1, wherein the specification is stored in a specification repository.

14. The system of claim 1, further comprising a specification extractor that queries a database for its schema and stores information associated with the schema in a specification repository.

15. A method of facilitating static checking of executable code comprising:

receiving executable code;

receiving a specification associated with the executable code, the specification comprising information associated with at least one of a precondition and a postcondition for a method;

statically applying the specification to the executable code;

determining whether a fault condition exists based, at least in part, upon the statically applied specification; and,

providing information associated with the fault condition, if a fault condition is determined to exist.

16. A computer readable medium having stored thereon computer executable instructions for carrying out the method of claim 15.

17. A method of developing a software component comprising:

implementing a subclass of a custom state class;

implementing at least one of a plug-in precondition and a plug-in postcondition as a method of the subclass;

placing a custom attribute on an enclosing type declaration that references the custom state sub class; and,

placing an attribute on a declaration that references the at least one of a plug-in precondition and a plug-in postcondition.

18. A computer readable medium having stored thereon computer executable instructions for carrying out the method of claim 17.

19. A method of performing static checking of executable code comprising:

invoking a precondition plug-in, providing the precondition plug-in with a program execution state;

receiving information from the precondition plug-in;

determining whether a fault condition exists based, at least in part, upon the information from the per-condition plug-in; and,

providing information associated with the fault condition, if a fault condition is determined to exist.

20. The method of claim 19, further comprising at least one of the following:

invoking a postcondition plug-in, providing the postcondition plug-in with the program execution state; and,

receiving information from the postcondition plug-in.

21. A data packet transmitted between two or more computer components that facilitates static checking of executable code, the data packet comprising:

a specification associated with executable code, the specification comprising information associated with at least one of a plug-in precondition and a plug-in postcondition, the specification providing information to be employed to statically check the executable code.

22. A computer readable medium storing computer executable components of an executable code check system comprising:

an input component that receives an object file and a specification associated with the object file, the specification comprising information associated with a plug-in condition for a method; and,

a checker component that employs the specification to facilitate static checking of the object file, the checker component providing information if a fault condition is determined.

23. An executable code check system comprising:

means for that receiving and a specification associated with the object file, the specification comprising information associated with a plug-in condition for a method; and,

means for statically checking the object file based, at least in part, upon the specification and determining if a fault condition exists; and,

means for providing information if a fault condition is determined to exist.

24. A method of performing static checking of executable code comprising:

receiving a request, the request including a parameter;

setting a type of a result of a method call to a type of the parameter; and,

employing the parameter only during static checking of the method.